Basic Specification

```
[VALUE]              - References memory at the specified index location. VALUE can
be supplied as an integer, or an integer variable.
_VALUE_              - References a variable with the specified name, VALUE.
increment VALUE    - Increments the integer value of VALUE.
decrement VALUE    - Decrements the integer value of VALUE.
repeat VALUE times {} - Repeats bracketed code VALUE number of times. VALUE can be
supplied as an integer literal, a memory pointer, or an integer variable.
set VALUEa to VALUEb - Assign VALUEb to VALUEa. VALUEa can be either a variable
reference or a memory reference. A variable or memory reference named VALUEa will be
allocated if it does not already exist. VALUEb can be any type. Multiple VALUEas can
be supplied.
output VALUE         - Print out VALUE to the debug console. VALUE can be a string
literal, an integer literal, a memory pointer, or a string or integer variable.
Multiple VALUEs can be supplied.
input to VARIABLE    - Get a string literal as input from the debug input box, and
assign it to VARIABLE.
```

Examples

1.

```
set [0] to 5;
set [2] to "Hello, World!";
set [1] to !true!;

increment [0];

output [0] [1] [2];
output "We will now clear all the values in memory, and try setting one
to a different value type.";

set [0] [1] [2] to NULL;
set [0] to "And for my next trick?";

output [0] "There we go!";
```

```
6
true
Hello, World!
We will now clear all the values in memory, and try setting one to a different value type.
And for my next trick?
There we go!
```

2.

```
set _num_ to 5;         // Declare a variable to say how many times to iterate.
set _value_ to 0;       // Declare a variable to keep track of our calls.

repeat _num_ times {    // Repeat the statements in brackets.
    increment _value_; // Increment the value we stored.
    output _value_;    // Display the value we have reached so far.
}

output "We have now finished a iterating. The result was:" _value_;
```

```
1
2
3
4
5
We have now finished a iterating. The result was:
5
```